REMARKS

Claims 1-8 and 11-17 are pending in this application, of which claims 1, 4, 7, 8, 13, 15 and 16 are independent.

To further prosecution, claims 4, 5, 7, 8, 11, 13, 15 and 16 are amended for clarity and to further distinguish over the cited prior art by adding limitations as suggested by the Examiner. For example, claims 4 and 5 are amended to include the limitations that the register file is grouped into two or more non-overlapping equally sized stacks of consecutive registers and that data hazard detect logic detects data hazards in the bypass data without differentiation between corresponding registers of each stack. Claim 7 is amended to add the limitation that each register ID aliases two or more non-consecutive rows of the register file. Claim 8 is amended to add the limitation that the register files is formed of equally sized non-overlapping groups of consecutive registers and that each register identifier aliases one corresponding register from each group of registers. Claim 13 is amended to include the limitation that one entry of the register ID file is aliased to two or more non-overlapping groups of sequential registers equivalent in size to the register ID file size. Claim 15 is amended to include the limitation that each register ID aliases non-consecutive registers of a register file. Claim 16 is amended to include the limitation that the stacked register file has two or more non-overlapping groups of consecutive registers and that each register ID aliases one register from each group.

Support for these amendments is found at least in FIG. 4 and paragraph [0022] of the immediate application.

Claims 9 and 10 are hereby cancelled.

Claim 17 is newly added and depends from claim 12, adding the limitation that each group of consecutive registers has 32 registers. Support for claim 17 is found in FIG. 4 and paragraph [0022] of the immediate application.

No new matter is added with the foregoing amendments.

Claim Rejections 35 U.S.C. § 102

Claims 8 and 15 stands rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,627,985 to Fetterman et al. (hereinafter "Fetterman"). Respectfully, we disagree.

By way of summary, the immediate application discloses register aliasing for data hazard detection logic of a processor that "simplifies the logic associated with ... data hazards so that a virtual register file may map frames of data to a physical register file of equal or larger size ... without corresponding growth of data hazard detect logic." See paragraph 11 of the immediate application. Complexity within the data hazard detection logic is reduced by aliasing multiple physical registers to one register ID within the data hazard detection logic. See paragraph 22 and FIG. 4. For example, FIG. 4 illustrates thirty-two register IDs, RID(32)-RID(63) within the data hazard detection logic that alias to one-hundred-and-twenty-eight general registers, GR(32)-GR(159), of the processor. Thus, in this example, each register ID within the data hazard detection logic is aliased to four general registers, such that the data hazard detection logic need only compare thirty-two register IDs to identify data hazards within any of one-hundred-and-twenty-eight general registers. To view this aliasing in another way, in this example, hazard detection logic aliases four general registers to one register ID thus matching any of the four general registers with each other as data hazards. The complexity of the data hazard detection logic is thereby reduced, since fewer register IDs are compared in comparison to the number of general registers.

On the other hand, Fetterman discloses a processor with a reorder circuit that has a plurality of physical registers, used to buffer speculative execution results, and a real register circuit with a plurality of committed state registers, used to buffer committed execution results. The out of order processor described by Fetterman, however, does not include hazard detection logic. In particular, the processor of Fetterman uses register renaming and a re-order buffer to provide a different approach to processing instructions and avoiding data hazards. More particularly, Fetterman's register renaming utilizes a one-to-one aliasing relationship between logical destination registers and physical destination registers. See Fetterman col. 5, lines 41-

46. Fetterman discloses that entries in the register alias table correspond to architectural registers of the original macroinstruction stream; it provides the example that EAX, EBX, ECX and EDX entries of the register alias table 80 correspond to the EAX, EBX, ECX and EDX register of the Intel Architecture Microprocessor, thus indicating a one-to-one renaming relationship. But Fetterman does not disclose hazard detection logic and therefore cannot teach one-to-many register aliasing in data hazard detection to simplify data hazard logic, as taught by the immediate application.

To anticipate a claim, Fetterman must teach every element of the claim and "the identical invention must be shown in as complete detail as contained in the ... claim." MPEP 2131 citing Verdegaal Bros. V. Union Oil Co. of California, 814 F.2d 628, 2 USPQ2d 1051 (Fed. Cir. 1987) and Richardson v. Suzuki Motor Co., 868 F.2d 1226, 9 USPQ2d 1913 (Fed. Cir. 1989). Fetterman does not teach every element of claims 8 and 15.

Amended claim 8 recites a method for data hazard detection within a processor, comprising:

a) aliasing each register identifier of a group of register identifiers to two or more registers of a register file of the processor, the register file formed of equally sized non-overlapping groups of consecutive registers and each register identifier aliasing one corresponding register of each group of registers; and

b) determining data hazards within the register file by comparing one or more of the register identifiers.

Step a) of claim 8 requires that each register identifier is aliased to one corresponding register of each equally sized non-overlapping group of consecutive registers of the register file. In the disclosure of Fetterman, Fetterman clearly discloses a one-to-one register mapping. For example, Fetterman discloses "each entry in the register alias table 80 contains a reorder buffer (ROB) pointer" that "specifies a physical register" and that "each entry in the register alias table 80 also contains a real register file valid (RRFV) flag" that indicates "whether the speculative result data for the corresponding architectural register has been retired to the

appropriate committed state register in the real register circuit 44." See Fetterman col. 8, lines 46-49. Thus, Fetterman discloses that each entry in the register alias table maps to one register in the real register file. Fetterman does **not** teach that one register identifier aliases to ***two or more*** registers of the register file, as required by step a). In particular, step a) requires that each register ID aliases one corresponding register of each group of registers. Fetterman makes no such disclosure.

Further, as noted above, Fetterman does not disclose hazard detection and therefore cannot anticipate step b).

For at least these reasons, Fetterman cannot anticipate claim 8. Reconsideration of claim 8 is respectfully requested.

Amended claim 15 recites a method of data hazard detection within a processor, including:

a) aliasing each register ID within data hazard detection logic to two or more non-consecutive registers of a register file; and

b) determining data hazards by matching register IDs within the data hazard logic.

Step a) of claim 15 requires that each register ID is aliased to two or more registers of a register file within data hazard detection logic. As argued above, Fetterman does not disclose hazard detection logic. Further, Fetterman does not disclose matching register IDs within the data hazard logic as required by step b). Further, again, Fetterman does not disclose hazard detection logic.

For at least these reasons, Fetterman cannot anticipate claim 15. Reconsideration of claim 15 is respectfully requested.

Claims 8-12, 15 and 16 stand rejected under U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,884,070 ("Panwar"). Respectfully we disagree.

Panwar discloses that "for single precision operations using aliased registers, there are at least four possible dependencies per instruction since each source register can have two possible dependencies." See Panwar col. 4, lines 11-14. Panwar does not disclose reducing complexity of hazard logic through use of aliasing. Rather, Panwar

discloses that "the single precision instruction, which could normally have up to four dependencies, is divided into two separate microinstructions wherein each microinstruction would have possibly only two data dependencies." See Panwar col. 6, lines 4-8. Panwar thus concerns aliasing of two single precision registers to one double precision register, and only provides for the specific case of single precision instructions. Panwar only reduces four dependencies to two dependencies for single precision instructions.

Amended claim 8 recites a method for data hazard detection within a processor, including:

a) aliasing each register identifier of a group of register identifiers to two or more registers of a register file of the processor, the register file formed of equally sized non-overlapping groups of consecutive registers, each of the register identifiers aliasing one corresponding register of each group of registers; and

b) determining data hazards within the register file by comparing one or more of the register identifiers.

Step a) of amended claim 8 requires that each register identifier of a group of register identifiers is aliased to two or more registers of a register file of the processor, the register file formed of equally sized non-overlapping groups of consecutive registers and each register identifier aliasing one corresponding register of each group of registers. On the other hand, Panwar only aliases consecutive registers and therefore cannot anticipate amended claim 8.

Claims 9 and 10 are cancelled.

Claims 11, 12 depend from claim 8 and benefit from like argument. Since Panwar cannot anticipate claim 8, Panwar cannot anticipate claims 11, 12.

Reconsideration of claims 8, 11 and 12 is respectfully requested.

Amended claim 15 recites a method of data hazard detection within a processor, including:

a) aliasing each register ID within data hazard detection logic to two or more non-consecutive registers of a register file; and

b) determining data hazards by matching register IDs within the data hazard logic.

Step a) of claim 15 requires that each register ID is aliased to two or more non-consecutive registers of a register file. Panwar does not disclose aliasing non-consecutive registers of a register file and therefore cannot anticipate claim 15.

Amended claim 16 recites a method for stacked register aliasing in data hazard detection logic of a processor, including:

a) aliasing two or more non-overlapping groups of consecutive registers of a stacked register file to one group of consecutive register IDs within the data hazard detection logic, each register ID aliasing one register from each group of consecutive registers; and

b) detecting data hazards, if any, associated with a first and second register of the stacked register file by comparing a first aliased register ID of the first register to a second aliased register ID of the second register within the data hazard detection logic.

Panwar does not disclose aliasing two or more non-overlapping groups of consecutive registers of a stacked register file to one group of consecutive register IDs within the data hazard detection logic, as required by step a) of claim 16. Specifically, step a) requires that each register ID aliases two or more non-consecutive registers. Panwar does not disclose aliasing non-consecutive registers. Panwar also does not disclose or suggest comparing register ID within hazard detection logic to detect data hazards, as required by step b).

For at least these reasons, Panwar cannot anticipate claim 16. Reconsideration of claim 16 is respectfully requested.

<u>Claim Rejections 35 U.S.C. § 103</u>

When applying 35 U.S.C. §103, the following tenets of patent law must be adhered to:

a) The claimed invention must be considered as a whole;
b) The references must be considered as a whole and must suggest the desirability and thus the obviousness of making the combination;
c) The references must be viewed without the benefit of impermissible hindsight vision afforded by the claimed invention; and
d) Reasonable expectation of success is the standard with which obviousness is determined. MPEP §2141.01, *Hodosh v. Block Drug Co., Inc.*, 786 F.2d 1136, 1134 n.5, 229 U.S.P.Q. 182, 187 n.5 (Fed. Cir. 1986).

In addition, it is respectfully noted that to substantiate a *prima facie* case of obviousness the initial burden rests with the Examiner who must fulfill three requirements. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the references or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on Applicant's disclosure. (emphasis and formatting added) MPEP § 2143, *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991)

Claims 4 and 5 are rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent Number 5,826,055 granted to Wang et al. (hereinafter "Wang") in view of Panwar. Respectfully we disagree.

Wang and Panwar, as with Fetterman, disclose an out-of-order processor that operates very differently from the processor of the immediate application. See the abstract of Panwar and col. 5 lines 23-32 of Wang. As argued above, such a processor avoids data hazards through use of register renaming and reorder buffers and thus does not require data hazard detection logic. Therefore, it would not be obvious to combine Wang and Panwar with regard to hazard detection in an explicitly parallel

instruction computing (EPIC) processor. Further, even when combined, Wang and Panwar do not render claims 4 and 5 obvious.

Amended claim 4 recites a processor for processing program instructions, including:

a) a register file grouped into two or more non-overlapping equally sized stacks of consecutive registers;

b) an execution unit having an array of pipelines for processing the instructions and for writing bypass data to the register file; and

c) data hazard detect logic for detecting data hazards in the bypass data without differentiation between corresponding registers of each stack.

Element a) of claim 4 requires that a register file be grouped into two or more non-overlapping equally sized stacks of consecutive registers. Neither Wang nor Panwar disclose a register file grouped into two or more non-overlapping equally sized stacks of consecutive registers. Element c) requires data hazard detect logic for detecting data hazards in bypass data without differentiation between corresponding registers of each stack. Again, Wang and Panwar do not disclose hazard detection logic. Although Wang discloses data dependency checking logic, this only identifies the dependency of one instruction upon another in determining required instruction execution order to avoid data hazards. See Wang col. 6, lines 57-62.

For at least these reasons, Wang and/or Panwar cannot render claim 4 obvious. Reconsideration of claim 4 is respectfully requested.

Claim 5 depends from claim 4 and benefits from like argument. However, claim 5 also includes other features that patentable distinguish over Wang and Panwar. For example, claim 5 recites a register ID file for facilitating data hazard detection, the register ID file having a plurality of register identifiers, the data hazard detect logic aliasing data hazard detection according to mapping of the register identifiers to corresponding registers of each stack. Neither Panwar nor Wang disclose register identifiers that map to corresponding registers of each stack; they cannot therefore render claim 5 obvious.

Reconsideration of claim 5 is respectfully requested.

Claim 7 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent Number 5,371,684 (hereinafter "Iadonato") in view of U.S. Patent Number 6,598,149 (hereinafter "Clift"). Respectfully we disagree.

Amended claim 7 recites that each register ID of the register ID file aliases row-to-row hazard detection of the register file by common data hazard detection logic for two or more non-consecutive rows of the register file. Claim 7 thus requires that each register ID alias non-consecutive rows of the register file. Neither Iadonato nor Clift disclose aliasing each register ID to non-consecutive rows of the register file within hazard detection logic. Iadonato does not teach or suggest – anywhere - aliasing or mapping register IDs. In fact, Iadonato discloses that "all source registers are compared with all previous destination registers for each instruction in window 102." See Iadonato col. 5, lines 23-25. Although Clift discloses register renaming, Clift does not disclose or suggest that each register ID alias non-consecutive rows of the register file.

For at least this reason, Iadonato and Clift, even when combined, cannot render claim 7 obvious. Reconsideration of claim 7 is respectfully requested.

Claims 13 and 14 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Panwar in view of U.S. Patent Number 5,706,478 (hereinafter "Dye"). Respectfully we disagree.

Panwar discloses use of micro-instructions to reduce the number of data dependencies from four to two per instruction. Panwar thus reduces the number of data dependencies that the processor is required to track for each instruction, but does not reduce the number of register-to-register dependencies to be tracked.

Dye makes no disclosure of hazard detection logic and does not disclose or suggest selection of a file size for a register ID file. The Examiner, in paragraph 16 of the pending office action, asserts that Dye provides Panwar with the capability to adapt to a greater storage capacity since Dye teaches of 128 registers. However, Dye does not overcome the shortfall of Panwar to reduce data hazard logic dependency on size of a register file. Specifically, if the register file size of Panwar increases from 32

to 128, the hazard detect logic of Panwar still increases since the use of microinstructions in Panwar only reduces instruction data dependency by a factor of two (i.e., from four dependencies per instruction to two per microinstruction). Therefore, even when combined, Panwar and Dye do not reduce data hazard logic dependency on size of the register file.

Amended claim 13 recites a method of reducing data hazard logic dependency on size of a register file within a processor, including:

a) selecting a register ID file size;

b) aliasing at least one entry of the register ID file to two or more registers of the register file, each of the two or more registers being located in a non-overlapping group of sequential registers equivalent in size to the selected register ID file size; and

c) evaluating matches between entries of the register ID file in the hazard logic without distinguishing between common aliased entries of the register file.

Neither Panwar nor Dye discloses selecting a register ID file size as required by step a) of claim 13. Although Dye discloses a <u>register file</u> with 128 registers, neither Panwar nor Dye disclose or suggest selecting a size for a <u>register ID file</u> as required by step a) of claim 13. The aliasing of Panwar relates to instruction addressing of the register file, and does not relate to data hazard logic. Step b) of claim 13 requires that each of the two or more registers aliased by each register of the register ID file be located in non-overlapping groups of sequential registers of the register file, each group equivalent in size to the selected register ID file size. Neither Panwar nor Dye disclose grouping of registers of a register file based upon selected size of a register ID file. For at least these reasons, even when combined, Panwar and Dye cannot render claim 13 obvious.

Claim 14 depends from claim 13 and benefits from like argument. However, claim 14 also has features that patentably distinguish over Panwar and Dye. For example, claim 14 recites that aliasing allows for increase of the register file size without a corresponding increase in data hazard detection logic. As argued above, any

increase in the register file size of Panwar would result in an increased number of register-to-register dependencies. The combination of Panwar and Dye cannot, therefore, render claim 14 obvious.

Examiner's indication of allowable subject matter is appreciated. In view of the above amendments and remarks, we solicit allowance of claims 1-8 and 11-17.

Applicants believe no fees are due in connection with this response. If any fee is due, please charge Deposit Account No. 08-2025.

Respectfully submitted,

By: Curt A. Vock

Curtis A. Vock, Reg. No. 38,356
LATHROP & GAGE L.C.
4845 Pearl East Circle, Suite 302
Boulder, CO 80301
Telephone: (720) 931-3011
Facsimile: (720) 931-3001